```
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMM MMM MMM MMM
SSS             UUU          UUU  MMM  MMM  MMM MMM
SSS             UUU          UUU  MMM   M   MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
```

FILEID**SUMLIST

```
  SSSSSSSS  UU      UU  MM      MM  LL            IIIIII     SSSSSSSS  TTTTTTTTTT
  SSSSSSSS  UU      UU  MM      MM  LL            IIIIII     SSSSSSSS  TTTTTTTTTT
SS          UU      UU  MMMM  MMMM  LL              II     SS              TT
SS          UU      UU  MMMM  MMMM  LL              II     SS              TT
SS          UU      UU  MM  MM  MM  LL              II     SS              TT
SS          UU      UU  MM  MM  MM  LL              II     SS              TT
  SSSSSS    UU      UU  MM      MM  LL              II       SSSSSS        TT
  SSSSSS    UU      UU  MM      MM  LL              II       SSSSSS        TT
        SS  UU      UU  MM      MM  LL              II             SS      TT
        SS  UU      UU  MM      MM  LL              II             SS      TT
        SS  UU      UU  MM      MM  LL              II             SS      TT          ....
        SS  UU      UU  MM      MM  LL              II             SS      TT          ....
SSSSSSSS    UUUUUUUUUU  MM      MM  LLLLLLLLLL    IIIIII     SSSSSSSS    TT          ....
SSSSSSSS    UUUUUUUUUU  MM      MM  LLLLLLLLLL    IIIIII     SSSSSSSS    TT          ....
```

```
LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0000     1          .TITLE  SUM$LIST
0000     2
0000     3          .IDENT  'V04-000'
0000     4
0000     5  ;
0000     6  ;*****************************************************************************
0000     7  ;*                                                                           *
0000     8  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     9  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000    10  ;*  ALL RIGHTS RESERVED.                                                     *
0000    11  ;*                                                                           *
0000    12  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    13  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    14  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    15  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    16  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    17  ;*  TRANSFERRED.                                                             *
0000    18  ;*                                                                           *
0000    19  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    20  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    21  ;*  CORPORATION.                                                             *
0000    22  ;*                                                                           *
0000    23  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    24  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    25  ;*                                                                           *
0000    26  ;*                                                                           *
0000    27  ;*****************************************************************************
0000    28  ;
0000    29
0000    30  ;++
0000    31  ; FACILITY:
0000    32  ;
0000    33  ; ABSTRACT:
0000    34  ;
0000    35  ;       Source Update Merge procedure to create update files list
0000    36  ;
0000    37  ;
0000    38  ; ENVIRONMENT:  USER MODE
0000    39  ;
0000    40  ; AUTHOR:       R. Newland
0000    41
0000    42  ;
0000    43  ; MODIFIED BY:
0000    44  ;
0000    45  ;
0000    46  ;       V03-001 BLS0175         Benn Schreiber          15-Jun-1982
0000    47  ;                               Add routine SUM$UPDATE_QUAL to process /UPDATE qualifier
0000    48  ;
0000    49  ;--
```

DECLARATIONS

```
                        0000     51          .SBTTL  DECLARATIONS
                        0000     52 ;
                        0000     53 ;
                        0000     54 ; Macro definitions
                        0000     55 ;
                        0000     56          DEFUPFBLK                       ; SUM update file block
                        0000     57          DEFSUMCBL                       ; SUM control block
                        0000     58          $CLIDEF                         ; CLI blocks
                        0000     59          $DSCDEF                         ; Descriptor definitions
                        0000     60 ;
                        0000     61 ;
                        0000     62 ;
                        0000     63 ;
                        0000     64 ; Local storage
                        0000     65 ;
                        0000     66 ;
                    00000000     67          .PSECT  SUM$RO_DATA,NOEXE,NOWRT,LONG
                        0000     68 ;
                        0000     69 ;
                        0000     70 ;
                        0000     71 SUM_INIAUDST:                           ; Initial audit string
2A 2A 57 45 4E 2A 2A 3B 00'     0000     72          .ASCIC  /;**NEW**/
                        08       0000
                        0009     73 ;
                        0009     74 ;
                    00000000     75          .PSECT  SUM$RW_DATA,NOEXE,LONG
                        0000     76 ;
                        0000     77 SUM_CLIVAL:                             ; Qualifier value descriptor block
                        0000     78          $CLIREQDESC -
                        0000     79                  RQTYPE = CLI$K_ASCIIVAL ; Convert an ASCII string
```

```
                        001C     81          .SBTTL  SUM$UPDATE_QUAL
                        001C     82 ;
                        001C     83 ;++
                        001C     84 ; Description:
                        001C     85 ;
                        001C     86 ; This procedure creates a singly linked update files list from the values of
                        001C     87 ; a /UPDATE qualifier.  The qualifier values are expected to be file specs.
                        001C     88 ; If no values are supplied with the qualifer a list containing one entry is
                        001C     89 ; created.
                        001C     90 ;
                        001C     91 ;
                        001C     92 ; Calling sequence:
                        001C     93 ;
                        001C     94 ;       CALLS/CALLG
                        001C     95 ;
                        001C     96 ;
                        001C     97 ; Inputs:
                        001C     98 ;
                        001C     99 ;        4(AP)  = Address of getvalue routine (normally CLI$GET_VALUE)
                        001C    100 ;        8(AP)  = Address of descriptor for qualifier name (normally 'UPDATE')
                        001C    101 ;       12(AP)  = address to put update files list address (the address
                        001C    102 ;                 of the first update file)
                        001C    103 ;       16(AP)  = Address of a dynamic string descriptor.  Its contents are
                        001C    104 ;                 indeterminate after this routine returns.
                        001C    105 ;
                        001C    106 ; Outputs:
                        001C    107 ;
                        001C    108 ;       None
                        001C    109 ;
                        001C    110 ; Implicit inputs:
                        001C    111 ;
                        001C    112 ;       The CLI data base
                        001C    113 ;
                        001C    114 ; Implicit outputs:
                        001C    115 ;
                        001C    116 ;       The update files list
                        001C    117 ;
                        001C    118 ; Procedure value:
                        001C    119 ;
                        001C    120 ;       SUM$_NORMAL =   Normal completion
                        001C    121 ;       LIB$_INSVIRMEM,
                        001C    122 ;       LIB$_BADBLOSIZ,
                        001C    123 ;       LIB$_BADBLOADR = Error from calling LIB$GET_VM or LIB$FREE_VM
                        001C    124 ;
                        001C    125 ;--
                        001C    126
                  00000000      127          .PSECT  SUM$CODE,NOWRT,LONG
                        0000    128
                  0DFC  0000    129          .ENTRY  SUM$UPDATE_QUAL,^M<R2,R3,R4,R5,R6,R7,R8,R10,R11>
                        0002    130
5B    10 AC     D0    0002    131          MOVL    16(AP),R11              ; R11 Points to the descriptor
          58    D4    0006    132          CLRL    R8                      ; Initialize file number
       0C BC    D4    0008    133          CLRL    @12(AP)                 ; Initialize update file listhead
          5B    DD    000B    134 10$:     PUSHL   R11                     ; Stack descriptor address for buffer
       08 AC    DD    000D    135          PUSHL   8(AP)                   ; Stack qualifier name string
04 BC     02    FB    0010    136          CALLS   #2,@4(AP)               ; Get next qualifier value
   59     50    D0    0014    137          MOVL    R0,R9                   ; Save error status
```

```
                58   D6  0017   138          INCL    R8                      ; bump file number
             52 5A   D0  0019   139          MOVL    R10,R2                  ; Save pointer to current block
           0118 30   1C  001C   140          BSBW    SUM_GETSUMBLK           ; Get memory for Update file block
          35 50   E9   001F   141          BLBC    R0,30$                  ; Error if LBC
             01 58   D1  0022   142          CMPL    R8,#1                   ; First time?
                04   12  0025   143          BNEQ    15$                     ; If NEQ no
          0C BC 5A   D0  0027   144          MOVL    R10,@12(AP)             ; Return address of first block
          32 59   E9   002B   145 15$:     BLBC    R9,40$                  ; Branch if no value
             6B   B5  002E   146          TSTW    DSC$W_LENGTH(R11)       ; Is there anything there?
                2E   13  0030   147          BEQL    40$                     ; If eql no, done
             01 58   D1  0032   148          CMPL    R8,#1                   ; Is this first time?
                03   13  0035   149          BEQL    20$                     ; If EQL yes
             62 5A   D0  0037   150          MOVL    R10,(R2)                ; Link this block with previous block
                      003A   151 ;
                      003A   152 ; Get memory block to save file specification
                      003A   153 ;
             24 AA   DF  003A   154 20$:     PUSHAL  UPF_Q_CMNT+4(R10)       ; Address to store memory address
             20 AA   DF  003D   155          PUSHAL  UPF_Q_CMNT(R10)         ; Address of word containing size
          00 BE   6B  3C  0040   156          MOVZWL  DSC$W_LENGTH(R11),@(SP) ; Store file spec size
    00000000'GF   02  FB  0044   157          CALLS   #2,G^LIB$GET_VM         ; Get virtual memory block
             09 50   E9   004B   158          BLBC    R0,30$                  ; Error if LBC
24 BA 04 BB 20 AA   28  004E   159          MOVC3   UPF_Q_CMNT(R10), -      ; Copy file specification from CLI area
                      0055   160                  @DSC$A_POINTER(R11), -  ; to virtual memory block
                      0055   161                  @UPF_Q_CMNT+4(R10)
                B4   11  0055   162          BRB     10$                     ; Loop until no more values
                      0057   163
    00000000'EF   00  FB  0057   164 30$:     CALLS   #0,SUM$LIB_ERR          ; Report library error
                07   11  005E   165          BRB     50$
    50 00848001   8F  D0  0060   166 40$:     MOVL    #SUM$_NORMAL,R0         ; Set successful finish
                04  0067   167 50$:     RET
```

```
                     .SBTTL  SUM$LIST_QUAL
                ;
                ;++
                ; Functional description:
                ;
                ;       This procedure creates a singly linked update files list
                ;       from the values of an /UPDATE qualifier.  The qualifier values
                ;       are expected to be file specifications.  If no values are
                ;       supplied with the qualifer a list containing one entry is
                ;       created.
                ;
                ;
                ; Calling sequence:
                ;
                ;       CALLS/CALLG
                ;
                ;       This procedure will normally be called from the process's
                ;       qualifier action routine for /UPDATE.
                ;
                ;
                ; Inputs:
                ;
                ;        4(AP)  = CLI call back address
                ;        8(AP)  = CLI parameter qualifer descriptor block address
                ;       12(AP)  = CLI work area address
                ;       16(AP)  = address to put update files list address (the address
                ;                 of the first update file)
                ;
                ;
                ; Outputs:
                ;
                ;       None
                ;;
                ; Implicit inputs:
                ;
                ;       The CLI data base
                ;
                ;
                ; Implicit outputs:
                ;
                ;       The update files list
                ;
                ; Procedure value:
                ;
                ;       SUM$_NORMAL =   Normal completion
                ;       LIB$_INSVIRMEM,
                ;       LIB$_BADBLOSIZ,
                ;       LIB$_BADBLOADR = Error from calling LIB$GET_VM or LIB$FREE_VM
                ;
                ;
                ;--
                ;
 05FC           .ENTRY  SUM$LIST_QUAL,^M<R2,R3,R4,R5,R6,R7,R8,R10>
                ;
 58   01   D0   MOVL    #1,R8                   ; Initialise file number
   00C7   30   BSBW    SUM_GETSUMBLK           ; Get memory for Update file block
 5A 50  E9     BLBC    R0,30$                  ; Error if LBC
```

```
                10 BC    5A    D0  0073   226          MOVL    R10,@16(AP)             ; Return address of first block
                                   0077   227 ;
                56    08 AC    D0  0077   228          MOVL    8(AP),R6                ; Get parameter qualifer parameter block
                      04 A6    B5  007B   229          TSTW    CLI$W_QDVALSIZ(R6)      ; Are there any values?
                         54    13  007E   230          BEQL    40$                     ; No if EQL, return with empty node
             57    00000000'EF  9E  0080   231          MOVAB   SUM_CLIVAL,R7           ; Set pointer to argument list
                            0E   11  0087   232          BRB     20$                     ; Got first block so skip
                                   0089   233 10$:
                   52    5A     D0  0089   234          MOVL    R10,R2                  ; Save block address
                         58     D6  008C   235          INCL    R8                      ; Increment file number
                      00A6      30  008E   236          BSBW    SUM_GETSUMBLK           ; Get memory block for next update file
                   39 50        E9  0091   237          BLBC    R0,30$                  ; Error if LBC
                   62    5A     D0  0094   238          MOVL    R10,(R2)                ; Link this block with previous block
                                   0097   239 20$:
                      08 AC     DD  0097   240          PUSHL   8(AP)                   ; Push parameters, qualifer block address
                      0C AC     DD  009A   241          PUSHL   12(AP)                  ; Work area address,
                         57     DD  009D   242          PUSHL   R7                      ; Qualifer value descriptor block
                04 BC    03     FB  009F   243          CALLS   #3,@4(AP)               ; Get next qualifier value
                                   00A3   244 ;
                                   00A3   245 ; Get memory block to save file specification
                                   00A3   246 ;
                      24 AA      DF  00A3   247          PUSHAL  UPF_Q_CMNT+4(R10)       ; Address to store memory address
                      08 A7      DF  00A6   248          PUSHAL  CLI$W_RQSIZE(R7)        ; Address of word containing size
             20 AA    00 BE      D0  00A9   249          MOVL    @(SP),UPF_Q_CMNT(R10)   ; Store file specification size
       00000000'GF       02      FB  00AE   250          CALLS   #2,G^LIB$GET_VM         ; Get virtual memory block
                      15 50      E9  00B5   251          BLBC    R0,30$                  ; Error if LBC
       24 BA    0C B7    20 AA    28  00B8   252          MOVC3   UPF_Q_CMNT(R10), -      ; Copy file specification from CLI area
                                   00BF   253                  @CLI$[_RQVALU(R7), -    ; to virtual memory block
                                   00BF   254                  @UPF_Q_CMNT+4(R10)
             C5 03 A7    00      E0  00BF   255          BBS     #CLI$V_MOREVALS, -      ; If more values go back for next
                                   00C4   256                  CLI$B_RQSTAT(R7),10$
          50    00848001 8F      D0  00C4   257          MOVL    #SUM$_NORMAL,R0 ; Indicate successful completion
                            07     11  00CB   258          BRB     40$
                                   00CD   259 30$:
   00000000'EF       00      FB  00CD   260          CALLS   #0,SUM$LIB_ERR          ; Report library error
                                   00D4   261 40$:
                            04     04  00D4   262          RET
```

```
                     00D5   264          .SBTTL  SUM$LIST_PARAM
                     00D5   265 ;
                     00D5   266 ;++
                     00D5   267 ; Functional description:
                     00D5   268 ;
                     00D5   269 ;       This procedure creates a singly linked update files list
                     00D5   270 ;       from the file specifications of a parameter.  If the
                     00D5   271 ;       parameter is null the list will be empty.
                     00D5   272 ;
                     00D5   273 ;
                     00D5   274 ; Calling sequence:
                     00D5   275 ;
                     00D5   276 ;       CALLS/CALLG
                     00D5   277 ;
                     00D5   278 ;       This procedure will normally be called from the program's
                     00D5   279 ;       parameter processing routines.
                     00D5   280 ;
                     00D5   281 ;
                     00D5   282 ; Inputs:
                     00D5   283 ;
                     00D5   284 ;        4(AP)  = CLI call back address
                     00D5   285 ;        8(AP)  = CLI parameter request descriptor block address
                     00D5   286 ;       12(AP)  = CLI work area address
                     00D5   287 ;       16(AP)  = address to put update files list address (the address
                     00D5   288 ;                 of the first update file)
                     00D5   289 ;
                     00D5   290 ;
                     00D5   291 ; Outputs:
                     00D5   292 ;
                     00D5   293 ;       None
                     00D5   294 ;;
                     00D5   295 ; Implicit inputs:
                     00D5   296 ;
                     00D5   297 ;       The CLI data base
                     00D5   298 ;
                     00D5   299 ;
                     00D5   300 ; Implicit outputs:
                     00D5   301 ;
                     00D5   302 ;       The update files list
                     00D5   303 ;
                     00D5   304 ; Procedure value:
                     00D5   305 ;
                     00D5   306 ;       SUM$_NORMAL =   Normal completion
                     00D5   307 ;       LIB$_INSVIRMEM,
                     00D5   308 ;       LIB$_BADBLOSIZ,
                     00D5   309 ;       LIB$_BADBLOADR = Error from calling LIB$GET_VM or LIB$FREE_VM
                     00D5   310 ;
                     00D5   311 ;
                     00D5   312 ;--
                     00D5   313 ;
                 057C 00D5   314          .ENTRY  SUM$LIST_PARAM,^M<R2,R3,R4,R5,R6,R8,R10>
                     00D7   315 ;
5A   10 AC    D0     00D7   316          MOVL    16(AP),R10              ; Get address to return Update list
       6A    D4     00DB   317          CLRL    (R10)                   ; Clear in case parameter is null
       58    D4     00DD   318          CLRL    R8                      ; Initialise file number
56   08 AC    D0     00DF   319          MOVL    8(AP),R6                ; Get request block address
                     00E3   320 10$:
```

```
                     00    DD   00E3    321          PUSHL   #0                      ; Call CLI to get next file spec
                     6E    9F   00E5    322          PUSHAB  (SP)
                  0C AC    DD   00E7    323          PUSHL   12(AP)                  ; Work area address
                     56    DD   00EA    324          PUSHL   R6                      ; Request block address
                04 BC      03   FB   00EC    325          CALLS   #3,@4(AP)               ; Call CLI
                   5E      04   C0   00F0    326          ADDL    #4,SP                   ; Tidy stack
             2E 03 A6      00   E1   00F3    327          BBC     #CLI$V_PARMPRS, -       ; Branch if no parameters
                                    00F8    328                  CLI$B_RQSTAT(R6),20$
                                    00F8    329 ;
                   52      5A   D0   00F8    330          MOVL    R10,R2                  ; Save block address
                           58   D6   00FB    331          INCL    R8                      ; Increment file number
                           38   10   00FD    332          BSB     SUM_GETSUMBLK           ; Get memory block for next update file
                        2D 50   E9   00FF    333          BLBC    R0,30$                  ; Error if LBC
                   62      5A   D0   0102    334          MOVL    R10,(R2)                ; Link this block with previous block
                                    0105    335 ;
                                    0105    336 ; Get memory block to save file specification
                                    0105    337 ;
                      24 AA     DF   0105    338          PUSHAL  UPF_Q_CMNT+4(R10)       ; Address to store memory address
                      08 A6     DF   0108    339          PUSHAL  CLI$W_RQSIZE(R6)        ; Address of word containing size
                20 AA    00 BE   D0   010B    340          MOVL    @(SP),UPF_Q_CMNT(R10)   ; Store file specification size
          00000000'GF    02   FB   0110    341          CALLS   #2,G^LIB$GET_VM         ; Get virtual memory block
                   15 50   E9   0117    342          BLBC    R0,30$                  ; Error if LBC
    24 BA    0C B6    20 AA   28   011A    343          MOVC3   UPF_Q_CMNT(R10), -      ; Copy file specification from CLI area
                                    0121    344                  @CLI$A_RQADDR(R6), -    ; to virtual memory block
                                    0121    345                  @UPF_Q_CMNT+4(R10)
             BD 03 A6      01   E0   0121    346          BBS     #CLI$V_CONCATINP, -     ; If more values go back for next
                                    0126    347                  CLI$B_RQSTAT(R6),10$
                                    0126    348 20$:
          50    00848001  8F   D0   0126    349          MOVL    #SUM$_NORMAL,R0 ; Indicate successful completion
                           07   11   012D    350          BRB     40$
                                    012F    351 30$:
          00000000'EF      00   FB   012F    352          CALLS   #0,SUM$LIB_ERR          ; Report library error
                                    0136    353 40$:
                           04   0136    354          RET
```

```
                                0137   356          .SBTTL  SUM_GETSUMBLK
                                0137   357
                                0137   358 ;++
                                0137   359 ;
                                0137   360 ; Functional description:
                                0137   361 ;
                                0137   362 ;       This routine obtains a memory block for an Update files block
                                0137   363 ;       and if successful initialises the block.
                                0137   364 ;
                                0137   365 ; Inputs:
                                0137   366 ;
                                0137   367 ;       R8  - File number
                                0137   368 ;
                                0137   369 ; Outputs:
                                0137   370 ;
                                0137   371 ;       R10 = Address of memory block
                                0137   372 ;
                                0137   373 ;--
                                0137   374 ;
                                0137   375 SUM_GETSUMBLK:
                          3C    BB    0137   376          PUSHR   #^M<R2,R3,R4,R5>
                00000000'EF     9F    0139   377          PUSHAB  SUM$VIRT_ADDR           ; Stack arguments for LIB$GET_VM
                00000000'EF     9F    013F   378          PUSHAB  SUM$BLSZE               ; Address of size of block
          00000000'GF    02     FB    0145   379          CALLS   #2,G^LIB$GET_VM         ; Get memory block
                    34 50      E9    014C   380          BLBC    R0,10$                  ; Error if LBC
             5A    0000'CF     D0    014F   381          MOVL    W^SUM$VIRT_ADDR,R10     ; Get block address
6A   0098 8F    00    6A    00     2C    0154   382          MOVC5   #0,(R10),#0,#UPF_K_BLN,(R10) ; Zero memory block
               0C AA    58     90    015C   383          MOVB    R8,UPF_B_FILENO(R10)    ; Store file number
            1C AA    28 AA     DE    0160   384          MOVAL   UPF_T_AUDST(R10), -     ; Set audit string descriptor
                                0165   385                  UPF_Q_AUDDS+4(R10)
         50    00000000'EF     9E    0165   386          MOVAB   SUM_INIAUDST,R0         ; Point to ASCIC initial audit string
                    51    80     9A    016C   387          MOVZBL  (R0)+,R1                ; Get length of ASCII part of string
 28 AA    60     51    28     016F   388          MOVC3   R1,(R0),UPF_T_AUDST(R10); Copy into block
       51    38 AA     9E    0174   389          MOVAB   UPF_T_NAM(R10),R1       ; Set NAM block pointer and
                                0178   390          $NAM_STORE NAM = R1, -          ; initialise as a NAM block
                                0178   391                  BID = #NAM$C_BID, -
                                0178   392                  BLN = #NAM$C_BLN
                50    01     D0    0180   393          MOVL    #1,R0                   ; Reset R0 (Cleared by MOVC5)
                                0183   394 10$:
                      3C     BA    0183   395          POPR    #^M<R2,R3,R4,R5>
                             05    0185   396          RSB
                                0186   397 ;
                                0186   398 ;
                                0186   399          .END
```

```
$CLI.                   = 00000000 R      03
$CLI..                  = 0000001C R      03
..AFLG                  = 00000000
..FLG                   = 00000002
..MOD                   = 00000000
..TYP                   = 0000001F
.LEN                    = 00000001
BIT...                  = 00000005
CLI$A_RQADDR            = 0000000C
CLI$B_RQSTAT            = 00000003
CLI$B_RQTYPE            = 00000000
CLI$C_REQDESC           = 0000001C
CLI$K_ASCIIVAL          = 00000041
CLI$L_RQVALU            = 0000000C
CLI$V_CONCATINP         = 00000001
CLI$V_MOREVALS          = 00000000
CLI$V_PARMPRS           = 00000000
CLI$W_QDVALSIZ          = 00000004
CLI$W_RQSIZE            = 00000008
DSC$A_POINTER           = 00000004
DSC$W_LENGTH            = 00000000
LIB$GET_VM                ********    X   04
NAM$B_BID               = 00000000
NAM$B_BLN               = 00000001
NAM$C_BID               = 00000002
NAM$C_BLN               = 00000060
NAM$K_BLN               = 00000060
SIZ...                  = 00000001
SUM$BLSZE                 ********      X   04
SUM$LIB_ERR               ********      X   04
SUM$LIST_PARAM            000000D5 RG       04
SUM$LIST_QUAL             00000068 RG       04
SUM$UPDATE_QUAL           00000000 RG       04
SUM$VIRT_ADDR             ********      X   04
SUM$_NORMAL             = 00848001
SUM_B_FLAGS               0000001C
SUM_CLIVAL                00000000 R        03
SUM_GETSUMBLK             00000137 R        04
SUM_INIAUDST              00000000 R        02
SUM_K_BLN                 0000001D
SUM_L_ISDATA              00000004
SUM_L_STS                 00000000
SUM_M_AUDIT             = 00000001
SUM_M_AUDITNEW          = 00000002
SUM_M_DELETE            = 00000010
SUM_M_SRCUPD            = 00000004
SUM_M_SUBCLSH           = 00000008
SUM_Q_AUDDS               00000008
SUM_Q_FILESP              00000010
SUM_V_AUDIT             = 00000000
SUM_V_AUDITNEW          = 00000001
SUM_V_DELETE            = 00000004
SUM_V_SRCUPD            = 00000002
SUM_V_SUBCLSH           = 00000003
SUM_W_INSERT_NO           0000001A
SUM_W_LINE_NO             00000018
UPF_B_EDFLAGS             00000009
UPF_B_FIFLAGS             00000008
UPF_B_FILENO              0000000C
UPF_K_BLN                 00000098
UPF_L_PTR                 00000000
UPF_Q_AUDDS               00000018
UPF_Q_CMNT                00000020
UPF_Q_EDITS               00000010
UPF_T_AUDST               00000028
UPF_T_NAM                 00000038
UPF_W_DOT                 0000000A
UPF_W_LOC1                00000004
UPF_W_LOC2                00000006
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000098 (152.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
SUM$RO_DATA	00000009 (9.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
SUM$RW_DATA	0000001C (28.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
SUM$CODE	00000186 (390.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	36	00:00:00.08	00:00:00.51
Command processing	135	00:00:00.53	00:00:02.28
Pass 1	241	00:00:07.06	00:00:16.03
Symbol table sort	0	00:00:00.77	00:00:02.23
Pass 2	79	00:00:01.49	00:00:03.34
Symbol table output	10	00:00:00.07	00:00:00.12
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	505	00:00:10.03	00:00:24.54

The working set limit was 1350 pages.
34417 bytes (68 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 535 non-local and 15 local symbols.
399 source lines were read in Pass 1, producing 26 object records in Pass 2.
32 pages of virtual memory were used to define 25 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SUM.OBJ]SUM.MLB;1	4
_255DUA28:[SYSLIB]STARLET.MLB;2	16
TOTALS (all libraries)	20

808 GETS were required to define 20 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:SUMLIST/OBJ=OBJ$:SUMLIST MSRC$:SUMLIST/UPDATE=(ENH$:SUMLIST)+LIB$:SUM/LIB
```

SUMLIST
LIS

SUMMAIN
LIS

SUMTFRVEC
LIS

SUMFILES
LIS

SUMMSG
LIS

SUMERROR
LIS

SUMOPEN
LIS

SYS

UPDATE
LIS

SYS
MAP